Dev 102

Dynamisez vos sites internet avec du Javascript

(Slides par Sami ‘Sfluor’ Tabet)

Plan

I - Introduction au JS

II - DOM et JS

III - Les requêtes HTTP

IV - Utilisation de l’API Youtube

I - Intro au JS

Multi-Paradigme: Script, Orienté Objet, impératif, fonctionnel

Typage: Faible, Dynamique

I - Intro au JS: Comment exécuter du JS

- Console Navigateur (outils de développement via Ctrl+Shift+I)
- Outils tels que jsbin, codepen
- NodeJS (cf Dev 103)
- Javascript intégré à du HTML

```
<script>
  console.log("Hello World ! ")
</script>
```

```
<script type="text/javascript" src="./myscript.js"></script>
```

I - Intro au JS: Opérateurs

```
3 + 4
// > 7

5 * 6
// > 30

17 / 3
// > 5.666666666666667

14 % 8
// > 6

14 // 8
// > 14

Math.floor(14/8)
// > 1
```

```
3 ** 4
// > 81

4 > 2
// > true

2 <= 2
// > true

3 == '3'
// > true

3 === '3'
// > false

1 === true
// > false
```

I - Intro au JS: Opérateurs

I - Intro au JS: Variables

```
// Déclarer une variable constante
const myConst = 'Shotgun';
// > undefined

// Tentative de modification
myConst = 'Paps';
// > TypeError: invalid assignment to const myConst

// Déclarer une variable variable
let myVar = 'Hello';
// > undefined

// Modification
myVar = 'Salut';
// > 'Salut'

// On vérifie
console.log(myVar);
// > 'Salut'
```

Toujours utiliser soit const, soit let lors des déclarations

I - Intro au JS: Variables le retour

```
// Déclarer une liste
const myList = ['ViaRézo', 42, false];
// > undefined

// Récupérer un élément
console.log(myList[1]);
// > 42

// Déclarer un objet
const myObject = {
    name: "Random GPA"
};
// > undefined

// Les objets ont leur attributs mutables
// même si définis avec const
myObject.age = 19;
// > 19

console.log(myObject.age);
// > 19
```

```
// Objet vide
const myEmptyObject = {};
// > undefined

// Second objet vide
const anOtherEmptyObject = {};
// > undefined

// Comparaison (ne pas utiliser ==
// si possible)
console.log(myEmptyObject == anOtherEmptyObject);
// > false

// Listes
const l1 = [1, 2, 3];
// > undefined

const l2 = [1, 2, 3];
// > undefined

// Même remarque
console.log(l1 == l2);
// > false
```

I - Intro au JS: Structures de Contrôle

```
const myVar = 2

// Condition
if (myVar === 1) {
    console.log("myVar value is 1");
}
else if (myVar === 3) {
    console.log("myVar value is 3");
}
else {
    console.log("myVar value is " + myVar );
}
// > "myVar value is 2"


// Boucle for
for (let i = 0; i < 10; i++) {
    console.log(i);
}
// > 1 ...


// Boucle while
while (condition) {
    // Do things
}
```

Alternatives aux boucles:

- map
- forEach
- filter
- reduce

I - Intro au JS: Fonctions

```
// Simple function
function multiply(a, b) {
    return a * b;
}

const result = multiply(2, 10);
// > undefined
console.log(result);
// > 20


// return is not mandatory
function sayHi(name) {
    console.log('Hello ' + name);
}

sayHi();
// > Hello

sayHi('John')
// > Hello John

// Other way to define functions
const add = function(a, b) {
    return a+b;
}

console.log(add(1, -1))
// > 0
```

```
// Functions can return functions
function createMultiplier(mult) {
    return function(number) {
        return mult * number;
    };
}

console.log(createMultiplier(5)(2));
// > 10

const multiplyBy5 = createMultiplier(5);
// > undefined

console.log(multiplyBy5(9));
// > 45


// Recursion also works
function fact(n) {
    if (n < 0) {
        return 0;
    }
    else if (n === 0) {
        return 1;
    }
    else {
        return n * fact(n-1);
    }
}

console.log(fact(5))
// > 120
```

I - Intro au JS: Un petit TP

- Faire une fonction qui détecte si un mot est un palindrome

- Faire une fonction qui calcule le n-ième terme de la suite de fibonacci

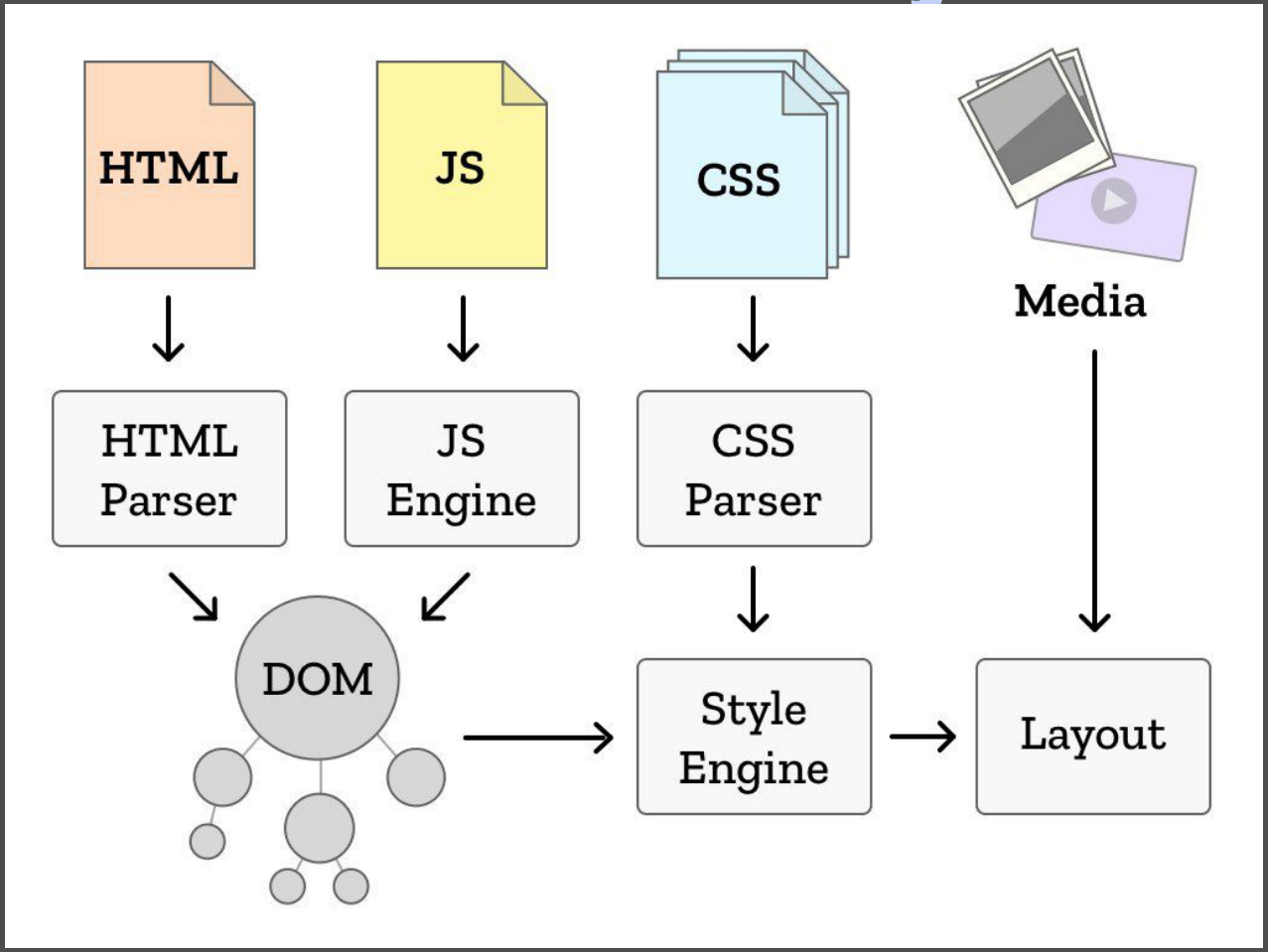
I - Intro au JS: Correction du TP

```
// Compute fibonacci's sequence
function fibonacci(n) {
    let a = 0;
    let b = 1;
    let result;

    for (let i = 0; i < n - 1; i++) {
        result = a + b;
        a = b;
        b = result;
    }

    return result;
}

function fibonacci2(n) {
    const fiveRoot = Math.sqrt(5);
    const phi = (1 + fiveRoot)/2;

    return (Math.pow(phi, n) - Math.pow(-1/phi, n))/fiveRoot;
}
```

```
// Check if a word is a palindrome
function isPalindrome(word) {
    const l = word.length;
    for(let i = 0; i < l; i++) {
        if (word[i] !== word[l-1-i]) {
            return false;
        }
    }
    return true;
}

function isPalindrome2(word) {
    return word === word.split('').reverse().join('');
}
```

II - DOM et Javascript: Rappels sur le DOM

DOM = Document Object Model



II - DOM et Javascript: les fonctions getElement

```html
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Get Elements</title>
</head>
<body>
  <div id="element1">Ceci est le premier élement</div>
  <p class="paragraph1">Ceci est le paragraphe 1</div>

  <script type="text/javascript">
    const element1 = document.getElementById('element1');

    console.log(element1.textContent);
    // > Ceci est le premier élément

    const paragraph1 = document.getElementByClassName('paragraph1');
    // > Ceci est le paragraphe 1
  </script>
</body>
</html>
```

- byId
- byName
- byClassName
- byTagName
- querySelector

II - DOM et Javascript: Création d'un noeud

```html
<!DOCTYPE html>
<html>
<head>
  <title>Creating nodes</title>
</head>
<body>

  <div id="mydiv">
    <h4> Paragraph is coming</h4>
  </div>

  <script>
    // Create a new paragraph node
    const para = document.createElement('p');

    // Prepare paragraph text
    const textNode = document.createTextNode('This is new.');

    // Append text to the newly created paragraph node
    para.appendChild(textNode):

    const element = document.getElementById('mydiv');
    element.appendChild(para);
  </script>
</body>
</html>
```

```html
<div id="mydiv">
    <h4>Paragraph is coming</h4>
    <p>This is new.</p>
</div>
```

node1.insertBefore(parent, node2)

parent.removeChild(child)

II - DOM et Javascript: Modifier un noeud

```html
<!DOCTYPE html>
<html>
<head>
  <title>Modifying nodes</title>
</head>
<body>

  <div id="data">
    Hi this text is gonna change
  </div>

  <script>

    const words = ['Salut', 'Bonjour', 'Shotgun', 'GPAs', 'GDAs'];

    // Returns a random item of a list
    function randomItem(list) {
      return list[Math.floor(Math.random()*list.length)];
    }

    // Randomly change data text
    function changeData() {
      const node = document.getElementById('data');

      node.textContent = randomItem(words);
    }

    // Apply a function every 2s
    setInterval(changeData, 2000);

  </script>
</body>
</html>
```

Pour modifier un attribut:

node.attribute = value

II - DOM et Javascript: les events listeners

```html
<!DOCTYPE html>
<html>
<head>
  <title>Creating nodes</title>
</head>
<body>

  <button class="alerting">
    Go go gadgeto alerte
  </button>

  <script>
  // Get the first button alerting
  var btn = document.getElementsByClassName('alerting')[0];

  // Add an event listener for the event click
  btn.addEventListener('click', alertUser)

  function alertUser() {
    alert('Vous avez cliqué sur le bouton !')
  }
  </script>
</body>
</html>
```

Events:

- mouseover
- mouseout
- change
- load
- keydown

II - DOM et Javascript: Un autre TP

Me: What type of exercice should we do

Me to me: A todolist

Me: But everybody is doing this

Me to me: Do it

II - DOM et Javascript: Un autre TP

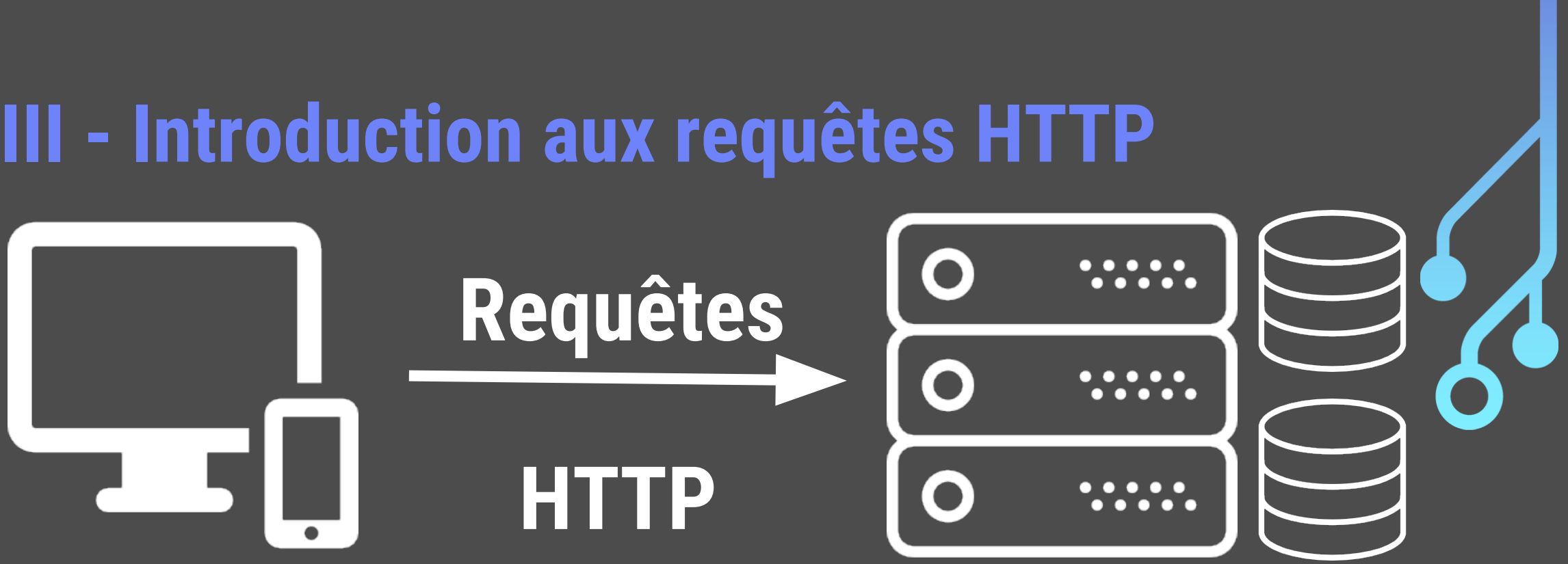
Faire une todolist:

- **Input + Bouton pour rajouter un todo**
- **Display de la list des todos**
- **Bonus: possibilité de retirer un todo**

II - DOM et Javascript: Correction

```html
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Todo List</title>
</head>
<body>
  <h1>Bienvenue sur la meilleure des todo lists</h1>
  <input id='todofield' />
  <button id='addtodo'>Ajouter ! </button>
  <ul id='todolist'>

  </ul>
  <script type="text/javascript">
    const todofield = document.getElementById('todofield');
    const addButton = document.getElementById('addtodo');
    const todolist = document.getElementById('todolist');

    function addTodo() {
        // Get input value
        const todo = todofield.value;

        // Wipe input
        todofield.value = '';

        // Create the new node
        const todoRow = document.createElement('li');
        const todoText = document.createTextNode(todo);
        todoRow.appendChild(todoText);

        // Append it to the list
        todolist.appendChild(todoRow);
    }

    // Listen for clicks
    addButton.addEventListener('click', addTodo);

  </script>
</body>
</html>
```

III - Introduction aux requêtes HTTP



Frontend:

- Visuel uniquement
- N'a pas accès aux BDDs
- Animé grâce au JS

Backend:

- S'occupe de la logique
- Interagit avec les BDDs
- (Peut générer du HTML/CSS)

III - Introduction aux requêtes HTTP

Types de requêtes principales:

- Get (requête pour récupérer une page ou des informations)
- Post (requête pour ajouter des données)
- Put (mise à jour de données)
- Delete (suppression de données)

III - Introduction aux requêtes HTTP

Description d'une requête get:

https://www.youtube.com/watch?v=DYMi21o4074

URL

Clé d'une variable

Valeur

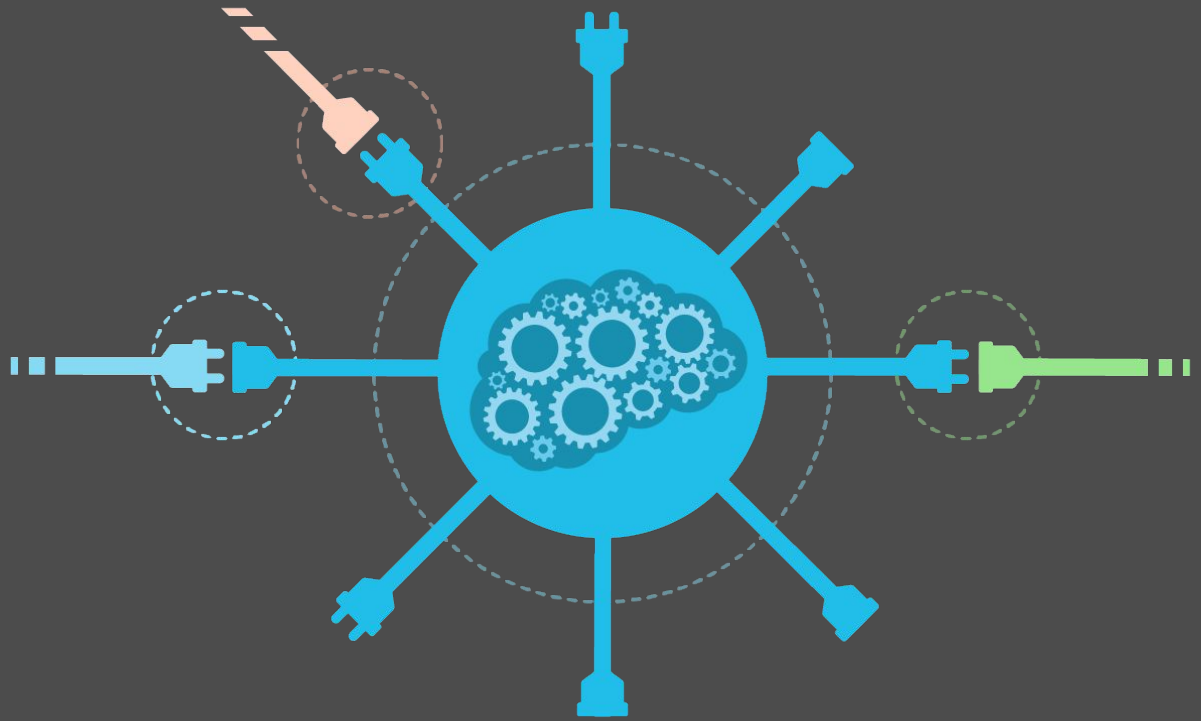
III - Introduction aux requêtes HTTP

Les requêtes HTTP en Javascript avec XMLHttpRequest

```
// On initialise la requête
const request = new XMLHttpRequest();
request.open('GET', 'https://noodle.viarezo.fr');

// On envoie la requête
request.send(null);

// On lit la réponse
console.log(request.responseText);
// > '<html>...'
```

IV - Une API ?

API = Application Programming Interface



IV - L'API Youtube

GET https://www.googleapis.com/youtube/v3/search
?part=snippet
&q=Happy
&key=[API_KEY]


```
{
  "items": [
    {
      "id": {
        "videoId": "y6Sxv-sUYtM"
      },
      "snippet": {
        "publishedAt": "2013-11-22T05:00:00.000Z",
        "title": "Pharrell Williams - Happy (Official Music Video)",
        "description": "Get Pharrell's album ",
      },
      "thumbnails": {
        "default": {
          "url": "https://i.ytimg.com/vi/y6Sxv-sUYtM/default.jpg",
          "width": 120,
          "height": 90
        }
      }
    },
    {
      "id": {
        "videoId": "JRMOMjCoR58"
      },
      "snippet": {
        "publishedAt": "2013-11-24T05:00:01.000Z",
        "title": "Pharrell Williams - Happy (1AM)",
        "description": "Get Pharrell's album G I R L",
      },
      "thumbnails": {
        "default": {
          "url": "https://i.ytimg.com/vi/JRMOMjCoR58/default.jpg",
          "width": 120,
          "height": 90
        }
      }
    },
  ]
}
```


IV - L'API Youtube: Exercice

Faire une app de recherche de vidéos:

http://tinyurl.com/dev102starter

- Input + Bouton pour récupérer les infos
- Display de la liste des informations
- Bonus: utiliser l'iframe YouTube pour intégrer la vidéo à notre site

```
<iframe src="http://www.youtube.com/embed/
ID_DE_LA_VIDEO?enablejsapi=1"></iframe>
```

IV - L’API Youtube: Correction

Correction de l’exercice:

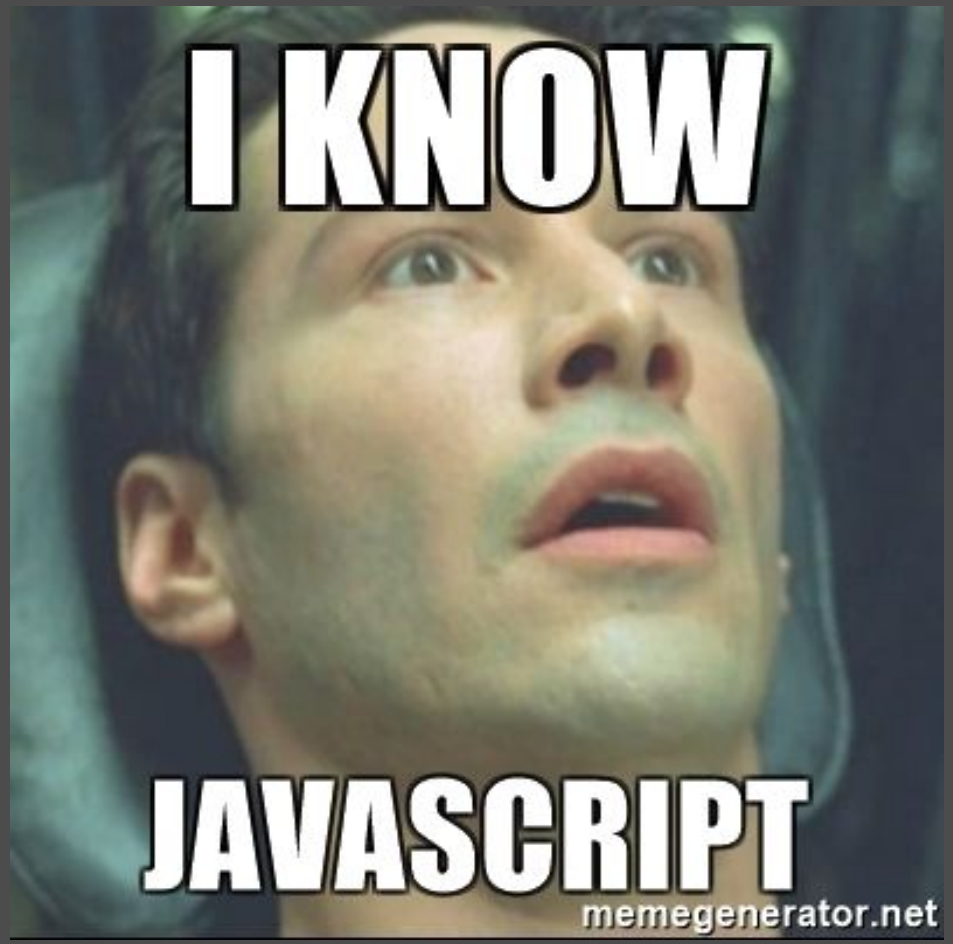
- http://tinyurl.com/dev102correction

Correction du bonus:

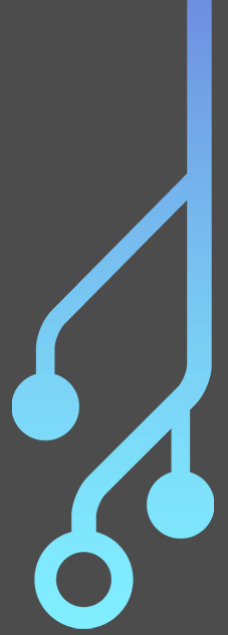
- http://tinyurl.com/dev102advancedcorrection

IV - L’API Youtube: Correction

V - Conclusion



V - Conclusion



Or not… :

- https://nodeschool.io
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/A_re-introduction_to_JavaScript
- https://reactjs.org/
- …